Lecture 23 ?

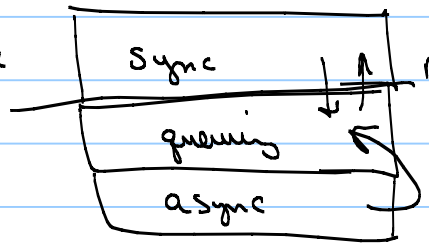
Note Title 4/15/2008

	Thursday	Thursday	
20	Loren Rasicka Nathan	Benjamin	10
10	Dave	Andy	10
10	Brian	Daniel	10
10	Brian	Damian Antunes	20
10	Nathaniel	Allen	10
10	Devin	Johnathan	10

Concurrency patterns – common ways of using concurrency.

Half-sync / Half-async } OS

App Space



read / write — unix I/O

old Unix ala 1977 - 1980 – fairly purely.

25 years of OS research

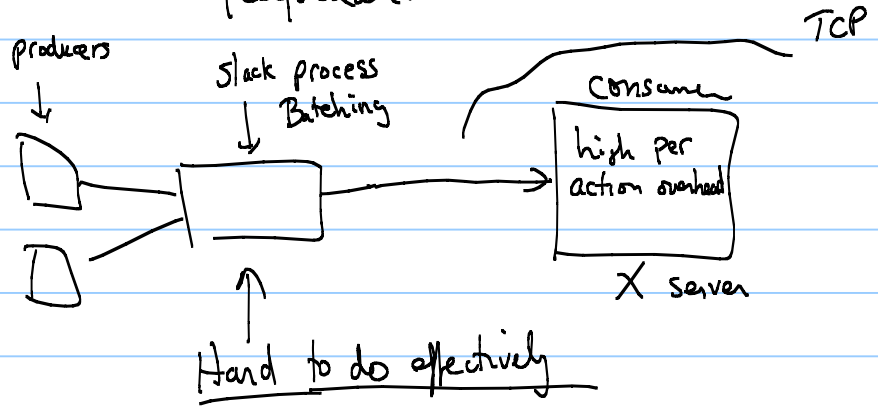
What are threads used for in a real system?

- Exploit concurrency – multi-processor or I/O
- Processor sharing – multiple applications making progress together
- Defer work when interactive response matters
  - do important work first
- Program structuring uses
  - Andrew Birrell: pipelines for exploiting concurrency
    - pipelines for program structure

      cat foo | grep bar | lpr

– "pumps" are components of pipelines

- Slack process — deliberately introduce some delay in a pipeline to improve overall performance.



Sleepers - 20% of threads

- wake up periodically and do a little work.
  - turn blinking cursor on and off. / garbage collector / file system state monitor
- problem: consume memory resources
- replace w/ a cron-like facility

One-shots - 7%

- get forked to watch for a specific occurrence

Mistake: programming by time

Wait with timeout : what should timeout be?
- UI stuff - externally governed
- processor, I/o, network operations - its unclear and changes w/ technology. } code maintenance issue.

timeouts can cover up bugs. — slow progress

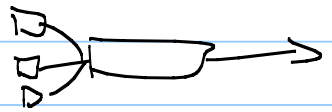
Dead lock avoidance: A before B before C 10%

Thread has equried locks A & C, now needs lock B

– fork a new thread to do the task requiring A, B, C.

Task Rejuvenation : uncaught exception at top level – 3%

Serializers : active monitors



Encapsulated forks:

Example: thread pool